Nuclear Application Programs Development and Integration for a Simulator

Hyun-Joon Park*, Tae-Woo Lee
KEPCO E&C, Inc., 989-111, Daedeok-dae-ro, Yuseong-gu, Daejeon, 34057, Korea
**Corresponding author: ko42971@kepco-enc.com*

1. Introduction

KEPCO E&C participated in the NAPS (Nuclear Application Programs) development project for BNPP (Barakah Nuclear Power Plant) simulator. The 3KEY MASTER™ was adopted for this project, which is comprehensive simulation platform software developed by WSC (Western Services Corporation) for the development, and control of simulation software.

The NAPS based on actual BNPP project was modified in order to meet specific requirements for nuclear power plant simulators. Considerations regarding software design for BNPP simulator and interfaces between the 3KM platform and application programs are discussed.

2. Methods and Techniques

In this section, software programming methods and techniques used to develop NAPS for BNPP simulator are described, especially based on empirical knowledge regarding the repeatability requirements of Nuclear Power Plant Simulators for Use In Operator Training and Examination [1]. Also, several ideas about software modification and migration from actual power plants to simulators are proposed.

2.1 Overview

3KEYMASTER (3KM) is used to develop full or partial scope simulators and provides task level execution rates. The 'task' is an execution unit that has own logical and/or arithmetic algorithms in order to replicate functions, conditions, and environments of actual plant systems. The task would be a kind of application programs configured in the 3KM environments. Tasks receive not only process values but also information including simulator's operational status parameters such as 'Reset/Run', 'Snap', 'Stop' and 'Backtrack', etc. The tasks are synchronized with schedules of the 3KM system and generally run 12 times per second. An execution frequency of tasks depends on specific 3KM task configurations.

NAPS for BNPP simulator is not registered as a task of the 3KM. Instead, NAPS is synchronized with a specific task, for example, in the Fig. 1 'TASK n', in order to interface with the 3KM indirectly. Consequently, NAPS receives information regarding the simulator status parameters and plant (simulator) process data via the 3KM task. Fig.1 shows a simple interface diagram between 3KM and NAPS.

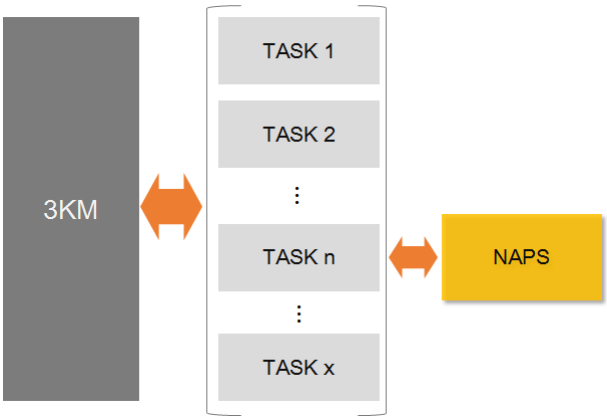

Fig. 1. Simple interface diagram between 3KM and NAPS

2.2 Considerations for repeatability requirements

The simulator testing requirements for repeatability state that "it shall be demonstrated that between successive simulator tests no noticeable differences exist with respect to time base relationships, sequences, durations, rate, and accelerations" in the section 4.1.1 of Nuclear Power Plant Simulators for Use In Operator Training and Examination [1].

In order to meet the repeatability requirement for nuclear power plant simulators, conventional application programs built for a reference plant would need to be modified.

In order to keep data stored in an application program, it is necessary to determine which variables in source codes need to be saved as an initial condition file. Purposes of the initial condition file are to store process data and conditions at specific point of time, and to have users retrieve the saved data when the users want to reset the simulator status.

The automatic local variables defined in functions typically don't need to be stored in the condition file because the automatic local variables usually hold temporary data or results in the middle point of whole calculation progresses, and they will be updated when the local functions are called.

In the case of global variables, they need to be saved into the condition file because they are normally used to preserve plant's status or calculated results throughout the lifetime of computer program. For the same reason, the static variable in local function should be stored into an initial condition data since the static variables hold results of previous calculation and are used for next algorithm calculation such as lag-filter, delay counter, etc.

If it is difficult to determine what variables need to be stored, one simple way is to store all or most of the variables defined in the source codes.

2.3 Recommendations regarding repeatability

Following recommendations should be considered when the conventional source codes are modified in order to meet the simulator repeatability requirement.

1) Construct new data structures to store variables that are needed to be saved into an initial condition file.

It is recommended to define new data structures replacing variables defined in dispersed source code files. This will make easier to maintain modified source codes in order to save simulator process data and status.

2) Replace local system time function calls with an incremental counter variable.

The simulator typically does not guarantee a constant execution interval in real time. For this reason, if the application software performs arithmetic or logic calculations depending on local system time, it will raise a possibility that produces non-repeatable manner.

3) Insert error-detecting code in order to ensure the condition file's integrity.

Insert an error-detecting code such as CRC-32 code into the condition file when the 'Snap' operation is performed. If errors occurred in the saved file for any reason, those errors or unintended changes to raw data can be detected by data integrity check codes.

2.4 Considerations for syncing external application programs

The execution of tasks registered in the 3KM configurations is managed by a scheduler of the 3KM.

In order to design and incorporate external application programs that are not a kind of the 3KM tasks, following ideas should be considered.

1) An external application program should be synced with a task of 3KM.

Both an external application and a 3KM task should be synced using synchronization APIs provided by system programming library. Then, the 'inner' task of 3KM will provide time slice needed to perform its algorithm of the external program.

The value of 'time slice' depends on 3KM settings. For example, the allocated time slice for the NAPS is 83.3 milliseconds because the 3KM task interfacing with NAPS runs 12 times per every second. Therefore, in order to show repeatable manner whenever user runs the simulator, external applications shall perform their functions during the allocated time slice.

An additional good practice for the synchronization among those programs is to design a task and external applications to include synchronization APIs for application calls and returns. If it is convinced that external applications perform their functions within the 'time slice', those synchronization techniques might not be needed.

Fig. 2 shows how the 3KM and NAPS interface as external application programs. The synchronization APIs were used to check a call and return of functions included in both side application programs.

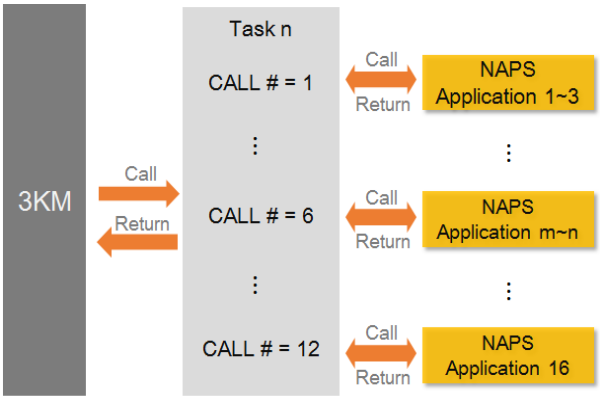

Fig. 2. Interface diagram for external applications

2) In order to sync heavy external applications with the 3KM, consider tracking time spent on execution of programs, and reducing CPU usage of processes.

If the external application exceeds the allocated time slice, the application may be divided into two or three parts in order to avoid impacting a system's performance.

The following programming tips are useful for enhancing the simulator's overall performance. Those can be applied not only external applications, but also direct tasks of 3KM.

- Refrain from writing unimportant log messages to disk files. The activities related to disk access usually take a lot of system resources.
- Remove unnecessary time delay function calls such as 'sleep' in legacy codes used for nuclear power plants.
- Turn on compiler's optimization flags to improve the software performance.

3. Conclusions

The repeatability is one of functional requirements for nuclear power plant simulators.

In order to migrate software from actual plants to simulators, software functions for storing and retrieving plant conditions and program variables should be implemented. In addition, software structures need to be redesigned to meet the repeatability, and source codes developed for actual plants would have to be optimized to reflect simulator's characteristics as well.

The synchronization is an important consideration to integrate external application programs into the 3KM simulator.

It is suggested that specific software modifications applied to simulators be reflected to software programs for actual plants, which decreases differences between both programs, and enhances software testability.

REFERENCES

[1] American National Standards Institute, Inc., "Nuclear Power Plant Simulators for Use In Operator Training and Examination", ANSI/ANS-3.5, p.7, 2009.